Final Report

Title: Implementation of a Novel Multiple Model Algorithm Based on Neural Network

Contract Number: F62562-03-P-0542

AFOSR/AOARD Reference Number: AOARD 03-41

AFOSR/AOARD Program Manager: Tae-Woo Park, Ph.D.

Period of Performance: 18 Sep 2003 – 17 Sep 2004

Submission Date: 08 Feb 2005

PI: Prof. Byung Ha Ahn
Gwangju Institute of Science and Technology

CoPI: Dr. Dae Bum Choi
Gwangju Institute of Science and Technology

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE **27 JUL 2006**	2. REPORT TYPE **Final Report (Technical)**	3. DATES COVERED **18-09-2003 to 18-01-2005**

4. TITLE AND SUBTITLE **Implementation of a Novel Multiple Model Algorithm Based on Neural network**	5a. CONTRACT NUMBER **F6256203P0542**
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) **Byung-Ha Ahn**	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Kwang-Ju Institute of Science and Technology (K-JIST),#1 Oryong-dong, Buk-gu,Gwang-Ju 500-712,KE,500-712**	8. PERFORMING ORGANIZATION REPORT NUMBER **AOARD-034041**
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) **The US Resarch Labolatory, AOARD/AFOSR, Unit 45002, APO, AP, 96337-5002**	10. SPONSOR/MONITOR'S ACRONYM(S) **AOARD/AFOSR**
	11. SPONSOR/MONITOR'S REPORT NUMBER(S) **AOARD-034041**

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT
: In this project, a multiple model method using neural network has been developed. The multiple model method is a technique where the uncertainty is not modeled in only one model. First, multiple models are designed for the uncertainty. Second, each model is scored or estimated using statistical methods. Based on the score of models, the most suitable model is selected.

15. SUBJECT TERMS
Target Tracking, Multiple Model Adaptive Estimation, Neural Networks

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**		**46**	

Abstract of the work

In this project, a multiple model method using neural network has been developed. The multiple model method is a technique where the uncertainty is not modeled in only one model. First, multiple models are designed for the uncertainty. Second, each model is scored or estimated using statistical methods. Based on the score of models, the most suitable model is selected.

Recently, the researches on the multiple model methods have focused on the dynamic multiple model methods that combine the multiple model method and Markov jump process (Markov chain). If the estimated model changes on time and the jump of system obeys the Markov jump process, the dynamic multiple model methods are very useful.

However, the weak point of the dynamic multiple model method is that the Markov transition probability matrix is not generally known. The Markov transition probability matrix defines the transition probability that a model at the previous time jumps to a model at the current time. In real applications, it is difficult to know the Markov probability transition matrix. As a result, without the correct Markov transition probability matrix, the dynamic multiple method can not achieve the best performance.

In this project, a new multiple model method without Markov jump process is developed. The neural network is substituted for Markov jump process with unknown transition probability matrix. Neural network is an artificial intelligence technique for search, classification and recognition based on the learning process. From the data that has implicitly the model transition, the neural network is trained for finding the information matched with the Markov transition probability matrix. Moreover, the developed method is verified using representative simulations.

List of Publications and Awards:

Publications before the project started:

[1] Daebum Choi, Byung-Ha Ahn, Hanseok Ko, 'Neural Net Baseb Variable Structure Multiple Model Reducing Mode Set Jump Delay', 11th IEEE workshop on Statistical Signal Processing, Singapore, 6~8 Aug. 2001.
[2] Daebum Choi, Monica Samal, Byung-Ha Ahn, 'On Neural-Net Based Variable Structure Multiple Model Method', The 6th World Multi-Conference on Systemics, Cybernetics and and Informatics, Orlando, 15~18 Aug. 2002.*
[3] Daebum Choi, Monica Samal, Byung-Ha Ahn. 'On Neural-Net Based Variable Structure Multiple Model Method', Journal of Systemics, Cybernetics, and Informatics, Vol. 1, No. 4, 2002.

* : This paper was selected as 5% best paper in the 6th World Multi-Conference on Systemics, Cybernetics and Informatics. This paper was published in Journal of Systemics, Cybernetics, and Informatics - See [3].

Now, we are preparing the journal paper based on the work in this project.

Index

Figures

Tables

1. Introduction

In target tracking, the state of moving object is inferred based on the sensor measurements. In the target tracking processes, two uncertainty factors are considered. One uncertainty factor is in the motion of the target. Mathematically, we can describe the motion of the target using the dynamic equation in the state space. For one dynamic equation, one movement can be modeled. Practically, it is impossible for a target to move based on a fixed dynamic equation. In this case, the real system that controls the movement of the target cannot realize the motions that are exactly modeled by the given mathematical dynamic equation. This is called the structural uncertainty [1]. The other uncertainty factor exists in the sensor system. All sensor systems have the instrumental errors. In other words, all the measurements are contaminated. Therefore, in target tracking, the unknown state of target is inferred based on the contaminated sensor measurements.

The estimation is to extract the information that we want to know from the data that are known and available. We can say the tracking process contains the state estimation of the target using sensor measurements. A popular tool for state estimation is the Kalman filter. In the Kalman filter, two uncertainty factors are modeled using the stochastic systems [2]. The stochastic model consists of the plant and measurement models. In the plant model, we can find the plant noise vector that describes the structural uncertainty. In the measurement model, there is the measurement noise vector that degrades the accuracy of measurements. If the plant and the measurement noise vector are suitably chosen, the Kalman filter estimates the state of the target reducing two uncertainty factors- the structural and the measurement uncertainty.

When the target maneuvers, the Kalman filter cannot optimally estimate the state of the target. When a target maneuvers, the dynamic equation used before maneuver is not in effect for state estimation. In other words, the structural uncertainty for the maneuvering movement cannot be covered by the plant noise. As a result, the wrong plant noise degrades the accuracy of the state estimation. One method for a maneuvering target is the multiple model (MM) methods, which seem to be the most promising [3].

In the MM methods, we prepare a set of stochastic systems that are describing various movements of the target and estimate the state of the target based on the results of the candidates. One stochastic system becomes one model. The main issue is how to combine the results of models in order to find the state estimate. One branch of MM researches is the dynamic(or switching) MM [1][4]. In the dynamic MM, the model change in time axis is assumed to be the Markov process. The dynamic MM has been developed intensively and the representative results are as follows: generalized pseudo-Bayesian (GPB) [1][4], interactive multiple model (IMM) [1][4] and variable structure interactive multiple model (VSIMM) [1][5]. The VSIMM method is the latest.

However, it is difficult to design the Markov transition probability matrix (MTPM) in the dynamic MM method. In early works, the MTPM was determined by designer [4]. There is no general rule for designing the MTPM and the MTPM is determined based on the designer's experience or trial and error. As a recent research on VSIMM, Doucet and Ristic developed a recursive MTPM estimation method [6]. However, consequently, the Markov jump property disappears in their work.

In this project, a new dynamic MM algorithm without Markov jump process is developed. The neural network is adopted instead of the Markov jump process.

The combination of target tracking and neural network has been studied. One issue is the neural network in the multiple target tracking. The aim is to use the Hopfield neural network for the Data association [7][8][9][10]. Other issue is to use neural network for fuzzy logic [11]. Moreover, neural network is used for compensating the error due to the maneuver of target. The Cho et. al proposed the adaptive Fuzzy tracker with error compensation using the neural network [12]. However, there is no research on combination of neural network and system model estimation in the dynamic MM methods.

This report is organized as follows. In Section II, the target tracking and Kalman filter are introduced. In Section III, the system model estimation in the dynamic MM with Markov process is summarized. In addition, the problem due to Markov process in the dynamic MM is presented.

In Section IV, the MM with neural network is proposed as a solution for the problem due to Markov process. The proposed method is developed based on the latest dynamic MM, VSIMM. In Section V, the proposed method is verified through 2 dimensional target tracking simulations. We compare the result of the proposed method and a VSIMM method. Finally, Section VI is devoted to conclusion.

2. Target Tracking and Kalman Filter

In this section, target tracking is explained and the uncertainty factors in target tracking are reminded. From the view point of the uncertainty in target tracking, Kalman filter is briefly reviewed.

2.1 Target Tracking

The process of inferring the state of an moving object (a target) based on the sensor measurements is called ``Target Tracking". In air-surveillance systems, it is important to know the position and the velocity of the aircraft of the interest. In order to infer the state of the target aircraft, we should prepare sensors, such as radar, which give the data on the state of the target aircraft. Based on the data from sensor, we determine the state of the target. Moreover, we use *a priori* information of target movement such as the maximum acceleration of target. We can mathematically design the dynamics of a known target. In summary, target tracking is a kind of data processing based on *a priori* information and the sensor measurement from the target. Figure 1 shows a target tracking system.

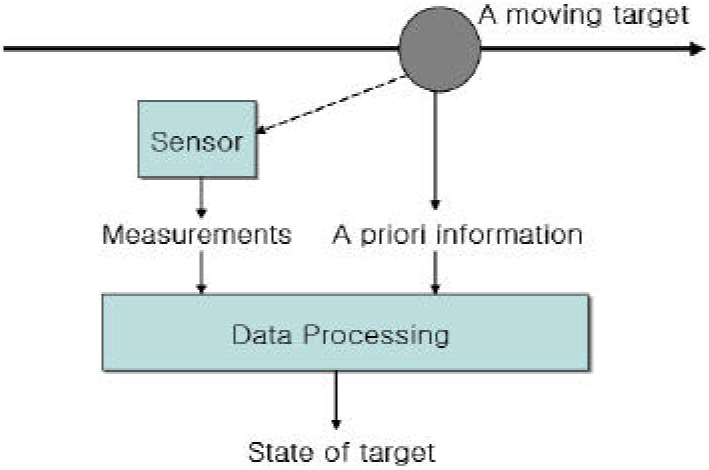


Figure 1 A Target Tracking System

The difficulty of target tracking arises from the uncertainty in the sensor data and the future movement of the target. In the next subsections, the uncertainty factors are explained.

2.1.1 Uncertainty in the target movement

Generally, it is very difficult to exactly know the future movement of the target. If the current state is exactly known and the target does not change its movement, we can predict the future motion of the target. In this case, we can mathematically express the motion of the target using the dynamic equations without uncertainty terms in the state space [2]. In practice, however, the exact state of the target is not known and we can not guarantee that the target keep its movement. In practice, we can only predict the state of the target with errors due to the incomplete information.

2.1.2 Uncertainty in the sensor measurements

It is impossible to eliminate noise from the measurements of practical sensor systems. The measurements from sensor are the only information that has the current status of the target. However, all the sensor systems are not free from the noise. Therefore, the error can not be eliminated and we should try to reduce the effect of the noise in the measurements.

2.2. Kalman Filter

The Kalman filter is the popular estimation tool that considers the two uncertainty factors. In the Kalman filter, the structural and the measurement uncertainty factors are treated in the plant and the measurement models, respectively. In this section, the Kalman filter is explained from the view point of the uncertainty factors in target tracking.

2.2.1 The System Model

The Kalman filter is built based on the system model composed of two models: the plant (dynamic) model and the measurement (observation) model. For the linear discrete model with Guassian noise, the system model can be formulated by:

$$x(k+1) = F(k)x(k) + G(k)v(k) \tag{2.1}$$

$$y(k) = H(k)x(k) + w(k) \tag{2.2}$$

where $x(k) \in \Re^n$ is the state vector, $x(k) \in \Re^m$ is the observation vector, $\Re^n$ is an n-dimensional Euclidean space, $v(k)$ and $w(k)$ are the zero-mean white Gaussian uncorrelated sequences, $v(k) \sim N(0, Q(k))$ and $w(k) \sim N(0, R(k))$, and k is the scan index. The initial state $x(0) \sim N(\bar{x}(0), P(0))$ is assumed to be uncorrelated with $v(k)$ and $w(k)$. In the plant model, the movement of the target is defined and in the measurement model, the process of measuring the information of the target movement is determined.

In the equations (2.1) and (2.2), the noise vectors $v(k)$ and $w(k)$ are the uncertainty factors as described in the previous section. The equation (2.1) without noise $v(k)$ is the dynamic equation that describes a motion of the target without external input. The noise term $v(k)$ is interpreted as the unknown variation of the state. By defining the $v(k)$, we can statistically model the uncertainty in the plant model. For example, the $v(k)$ is assumed to be a scalar zero-mean unit-variance Gaussian, then the variation of the state is predicted to be in the interval $(-3,3)$ with a confidence of about 99%. The equation (2.2) without the measurement noise $w(k)$ is the formula that transforms the state of the target to the sensor measurement. In the real sensor systems, the sensor measurement is contaminated due to various factors such as the weather, the hostile environment and the limit of the sensor systems. The contamination of the measurements are statistically modeled as the noise $w(k)$. In other words, the accuracy of the measurement depends on the measurement noise $w(k)$.

2.2.2 Kalman Filter Update Process

Based on the system model, the Kalman filter estimates the state of the target based on the following equations [2],[4]:

$$\hat{x}(k \mid k-1) = F(k)\hat{x}(k-1 \mid k-1), \hat{x}(0 \mid 0) = \overline{x}(0) \tag{2.3}$$

$$P(k \mid k-1) = F(k-1)P(k-1 \mid k-1)F^{T}(k-1) + G(k-1)Q(k-1)G^{T}(k-1) \tag{2.4}$$

$$K(k) = P(k \mid k-1)H^{T}(k)\left[H(k)P(k \mid k-1)H^{T}(k) + R(k)\right]^{-1} \tag{2.5}$$

$$\hat{x}(k) = \hat{x}(k \mid k-1) + K(k)\left[y(k) - H(k)\hat{x}(k+k-1)\right] \tag{2.6}$$

$$P(k \mid k) = \left[I_n - K(k)\right]P(k \mid k-1), P(0 \mid 0) = P(0) \tag{2.7}$$

where $P(k \mid k) = \text{cov}\{\tilde{x}(k), \tilde{x}(k)\}$, $\tilde{x}(k) = x(k) - \hat{x}(k \mid k)$, and I_n is the $n \times n$ unit matrix.

In the equation (2.3), the state of target at scan k is predicted based on the plant model with the information until scan k-1. After the state prediction, the error covariance of the state is predicted based on the state prediction and the plant model, which is given in the equation (2.4). In the next step, the Kalman gain is calculated based on the state and the error covariance predictions, which is given in the equation (2.5). Finally, we obtain the state and the error covariance estimations in the equation (2.6) and the equation (2.7), respectively.

In Kalman filter update process, let's analyze the role of Kalman gain. In the equation (2.4), the predicted error covariance is a function of previous error covariance and the plant noise variance $Q(k)$. As the $Q(k)$ increases, the Kalman gain increases. In the equation (2.5), the Kalman gain is calculated. As the error covariance $P(k \mid k-1)$ increases, the Kalman gain increases but as the measurement noise covariance increases, the Kalman gain decreases. As a result, Kalman gain increases when the plant noise increases and the measurement noise variance decreases. In the equation (2.6), the state estimation is obtained by addition of the state estimation and Kalman gain multiplied by the measurement innovation that is defined as the difference between the current measurement $y(k)$ and the measurement prediction, $H(k)\hat{x}(k \mid k-1)$. As the Kalman gain increases, the measurement innovation have more influence on the state estimate. The physical meaning of the measurement innovation is the error between the current measurement and the measurement prediction based on the previous state prediction. When the Kalman gain increases, we add more error to the state estimation and the updated measurement becomes dominant in current state estimation and the effect of the state prediction decreases. Remind that the Kalman gain increases when the plant noise variance increases and the measurement noise variance decreases. The meaning of big plant noise variance is that the plant model has big uncertainty. Intuitively, the state prediction is not reliable when the plant model has big uncertainty. Therefore, when the big Kalman gain reflects the big uncertainty in the plant model and the effect of the state prediction based on the plant model decreases. Moreover, as the measurement noise variance is smaller and smaller, the sensor measurement is more and more accurate. Intuitively, it is desirable that the updated measurement has a strong influent on the current state estimate when the measurement is accurate. Since the smaller measurement noise variance leads the bigger Kalman gain, the updated measurement has more influence on the current state estimate.

2.2.3 Problems in the Kalman Filter for the Maneuvering Target

When a target maneuvers, the state estimate of Kalman filter becomes erroneous. When the target changes its motion (dynamics) over the limit of the process noise, we call it `maneuver'. When a target maneuvers, the plant model should be changed. If we can not know the correct plant model and the incorrect plant model is used in the Kalman filter update, the state estimate is predicted to be erroneous.

In the next section, the MM methods are introduced as a solution for the maneuvering target tracking.

3. The Multiple Model Methods

In this section, the MM method is introduced for the maneuvering target tracking. First the main idea of the MM methods is explained. Next, the idea of the MM is expanded to the MM with Markov jump process and the VSIMM method. In addition, the weak point of the MM with the Markov jump process is explained at the end of this section.

3.1 The Main Idea of the MM Method

The key in the maneuvering target tracking is how to infer the plant model for the maneuvering target. As explained in the previous chapter, it is difficult to know the plant model when the target maneuvers. With the incorrect plant model, the state estimate is predicted to be erroneous. In the MM methods, we prepare a set of the system models where the various movements of the target are formulated. During the target tracking, each system model gives its own estimate. After obtaining all estimate from all the system models, we score the system models and combine the state estimates based on the scores.

We design n discrete system models. Let m_i denote i-th system model in the total system model set, $M = \{m_i\}$ for i=*1,…,n*. The $m_i(k)$ means a system model m_i is at scan k. The score of $m_i(k)$ is defined as $b_i(k)$. The local state estimate based on $m_i(k)$ is defined as $\hat{x}_i(k)$. The overall state estimate is given by:

$$\hat{x}(k) = \sum_{i=1}^{n} b_i(k)\hat{x}_i(k) \qquad (3.1)$$

$$\sum_{i=1}^{n} b_i(k) = 1 \qquad (3.2)$$

The MM is the process of calculating the scores, $b_1(k),\ldots,\ b_n(k)$ and the state estimate of the system models, $\hat{x}_1(k),\ldots,\ \hat{x}_n(k)$. In this project, the process of scoring system models is called system model estimation.

We can classify the MM methods based on the interaction among the system models [1],[4]. If there is no interaction among the system models, we call it the static MM, otherwise the dynamic MM. The mainstream of the MM researches is the dynamic MM methods such as IMM [1],[4] and VSIMM [1],[5].

3.2The Static MM Methods

In the static MM [4], the score of a system model is given by:

$$b_i(k) = P\{m_i(k) \mid y^k\} \tag{3.3}$$

where y^k is the set of the accumulated measurements until scan k, $y_1, \ldots, y_k$.

In the system model estimation of the static MM, the score of a model at scan k, $m_i(k)$ is not effected by the other models at scan *k-1*, but the only itself at scan *k-1*, $m_i(k-1)$. This means no interactions in the system models. The equation (3.3) [4] is rewritten as

$$b_i(k) = P\{m_i(k) \mid y^k\} = \frac{p\left(y(k) \mid y^{k-1}, m_i(k-1)\right) b_i(k-1)}{\sum_{j=1}^{n} p\left(y(k) \mid y^{k-1}, m_i(k-1)\right)} \tag{3.4}$$

where $p(\cdot)$ denotes probability density function. The term $p\left(y(k) \mid y^{k-1}, m_i(k-1)\right)$ is the likelihood of the current measurement, $y(k)$. In the equation (3.4), the denominator is a normalization of the likelihood. As a result, the score of the system model, $m_i(k)$, is effected by the likelihood of the current measurement and the past own status, $b_i(k-1)$.

3.3 The Dynamic MM Methods

In the dynamic MM, there are interactions among the system models. This interaction is built up based on the Markov jump process. In this subsection, the Markov jump process is briefly explained and the dynamic MM methods are introduced.

3.3.1 Markov Jump Process

In the Markov process [4,6], the current state is known from the most recent past state that we can know. The Markov process is defined by:

$$P\{q(t_k) \mid q(t), t \le t_{k-1}\} = P\{q(t_k) \mid q(t_{k-1})\} \tag{3.5}$$

where $q(t_k)$ is the state at time t_k.

The Markov jump process (Markov chain) is a Markov process having the finite number of states, $q_1, \ldots, q_n$. A Markov jump process is characterized based on the state transition probabilities given by:

$$p_{q_i,q_j} = P\{q_i(t) \mid q_j(t-1)\} \tag{3.6}$$

and the Markov transition probability matrix(MTPM) is given by:

$$\Pi = \{p_{q_i,q_j}\}. \tag{3.7}$$

The change of states on time is modeled by using the MTPM in the Markov jump process.

3.3.2 The Dynamic MM methods

In the dynamic MM methods, the interaction among the system models are built based on the Markov jump process. In this case, we consider the system model in the dynamic MM as the state in Markov jump process. Moreover, the transition from one system model to the other model is defined based on the MTPM.

In the system model estimation of the dynamic MM method, the history (or sequence) of the system model transition should be considered. Let $I(k) = \{m(1), \mathrm{K}\ , m(k) \mid m(1), \mathrm{K}\ , m(k) \in M\}$ be a sequence of the system models until scan k. Assume that $I_{m_i,m_j}(k) = \{m_i(k), I_{m_j}(k-1)\}$ and $I_{m_j}(k-1) = \{m_j, I(k-2)\}$ for $k > 2$. Define $b_{i,j}(k)$ as a new score of the system model transition from $m_j(k-1)$ to $m_i(k)$ [4], which is give by:

$$
\begin{aligned}
& b_{i,j}(k) \\
& \equiv b_{I_{m_i,m_j}(k)}(k) \\
& = P\{I_{m_i,m_j}(k) \mid y^k\} \\
& = P\{I_{m_i,m_j}(k) \mid y(k), y^{k-1}\} \\
& = \frac{1}{c} p\left(y(k) \mid I_{m_i,m_j}(k), y^{k-1}\right) P\{I_{m_i,m_j}(k) \mid y^{k-1}\} \\
& = \frac{1}{c} p\left(y(k) \mid I_{m_i,m_j}(k), y^{k-1}\right) P\{m_i(k), I_{,m_j}(k-1) \mid y^{k-1}\} \\
& = \frac{1}{c} p\left(y(k) \mid I_{m_i,m_j}(k), y^{k-1}\right) P\{m_i(k) \mid I_{,m_j}(k-1), y^{k-1}\} b_{I_{m_j}(k-1)}(k-1) \\
& = \frac{1}{c} p\left(y(k) \mid I_{m_i,m_j}(k), y^{k-1}\right) P\{m_i(k) \mid I_{,m_j}(k-1)\} b_j(k-1) \\
& = \frac{1}{c} p\left(y(k) \mid I_{m_i,m_j}(k), y^{k-1}\right) P\{m_i(k) \mid m_j(k-1), I(k-2)\} b_j(k-1) \\
& = \frac{1}{c} p\left(y(k) \mid I_{m_i,m_j}(k), y^{k-1}\right) P\{m_i(k) \mid m_j(k-1)\} b_j(k-1) \qquad (3.8)
\end{aligned}
$$

where c is a normalization constant. Since the system model transition is defined based on the Markov jump process, the sequence $I(k-2)$ in $P\{m_i(k) \mid m_j(k-1), I(k-2)\}$ can be ignored. Using Markov jump process, the $b_{i,j}(k)$ is rewritten by:

$$b_{i,j}(k) = \frac{1}{c} p\left(y(k) \mid I_{m_i,m_j}(k), y^{k-1}\right) p_{i,j} b_j(k-1). \qquad (3.9)$$

In the equation (3.9), it is clear that the system model $m_j(k-1)$ have influence on the system model $m_i(k)$. The score $b_{i,j}(k)$ is a function of the likelihood of the updated measurement, $p\left(y(k) \mid I_{m_i,m_j}(k), y^{k-1}\right)$, the Markov state transition probability, $p_{i,j}$, and the previous score of a system model, $b_j(k-1)$. The Markov jump process scores the effect from the previous system model to the current. As a result, the previous system model effects the current system model estimation through Markov jump process.

Finally, the score of the dynamic MM methods are given by:

$$b_i = \sum_{j=1}^{n} b_{i,j} \,. \tag{3.10}$$

3.4 Variable Structure Interactive Multiple Model Method

In this subsection, the recent and advanced dynamic MM method, VSIMM is introduced. The VSIMM and the IMM are compared and the system model estimation in VSIMM is explained.

3.4.1 The Difference of IMM and VSIMM}

The VSIMM has been developed in order to solve a dilemma in the IMM [1,5]. In the IMM methods, as the number of models increases, the confliction among models causes increasing estimation error. In the equation (3.9), the score of the system model at scan k is the sum of the weighted scores of all the system models. However, the true system model is not expressed in terms of all the system models but the limited number of the system models. As a result, the system models except for the limited number of models that express the true system model become the error terms. However, when we use small number of models, tracking performance will be degraded. If a target maneuvers and its movements cannot be covered with the small size of mode set, the state estimation will be erroneous.

In order to solve dilemma in the IMM, the VSIMM selects the system models used for the state estimation [1,5]. In this case, we add the system model selection method to the state estimation of the IMM. If all system models are selected for the system model estimation, the VSIMM and IMM are same [1,5].

3.4.2 System Model Estimation in VSIMM

Let $s(k) \subset M$ denote a system model set, whose elements are the selected system models for the state estimation, at scan k and $S(k) = \{s(1), \mathrm{K}\ , s(k)\}$ is the selected system model set sequence until scan k. Let $m_{s(k)}$ denote a system model in the selected system model set, $s(k)$. Let $H(k) = \{m(1), \mathrm{K}\ , m(k) \mid m(1) \in s(1), \mathrm{K}\ , m(k) \in s(k)\}$ be a sequence of the selected system model. Define $H_{m_i, m_j}(k) = \{m_i(k), H_{m_k}(k-1) \mid m_i(k) \in s(k)\}$ as a sequence of the selected system model with $m_i(k)$ and $m_j(k-1)$, where $H_{m_j}(k-1) = \{H(k-2), m_j(k-1) \mid m_j(k-1) \in s(k-1)\}$.

The $b_{i,j}(k)$ for the VSIMM is give by:

$$
\begin{aligned}
& b_{i,j}(k) \\
& \equiv b_{I_{m_i, m_j}(k)}(k) \\
& = P\{H_{m_i, m_j}(k) \mid y^k\} \\
& = P\{H_{m_i, m_j}(k) \mid y(k), y^{k-1}\} \\
& = \frac{1}{c} p\left(y(k) \mid H_{m_i, m_j}(k), y^{k-1}\right) P\{H_{m_i, m_j}(k) \mid y^{k-1}\} \\
& = \frac{1}{c} p\left(y(k) \mid H_{m_i, m_j}(k), y^{k-1}\right) P\{m_i(k), H_{, m_j}(k-1) \mid y^{k-1}\} \\
& = \frac{1}{c} p\left(y(k) \mid H_{m_i, m_j}(k), y^{k-1}\right) P\{m_i(k) \mid H_{, m_j}(k-1), y^{k-1}\} b_{I_{m_j(k-1)}}(k-1) \\
& = \frac{1}{c} p\left(y(k) \mid H_{m_i, m_j}(k), y^{k-1}\right) P\{m_i(k) \mid H_{, m_j}(k-1)\} b_j(k-1) \\
& = \frac{1}{c} p\left(y(k) \mid H_{m_i, m_j}(k), y^{k-1}\right) P\{m_i(k) \mid m_j(k-1), H(k-2)\} b_j(k-1) \\
& = \frac{1}{c} p\left(y(k) \mid H_{m_i, m_j}(k), y^{k-1}\right) P\{m_i(k) \mid m_j(k-1)\} b_j(k-1)
\end{aligned}
\tag{3.11}
$$

where c is a normalization constant. Since the system model transition is defined based on the Markov jump process, the sequence $H(k-2)$ in $P\{m_i(k) \mid m_j(k-1), H(k-2)\}$ can be ignored. Using Markov jump process, the $b_{i,j}(k)$ is rewritten by:

$$b_{i,j}(k) = \frac{1}{c} p\left(y(k) \mid H_{m_i,m_j}(k), y^{k-1}\right) p_{i,j} b_j(k-1) \ . \tag{3.12}$$

Finally, the score of the VSIMM methods are given by:

$$b_i = \frac{\sum_{j=1}^{n} b_{i,j}}{\sum_{i=1}^{n} \sum_{j=1}^{n} b_{i,j}} \tag{3.13}$$

There is an important assumption in VSIMM: the admissible property [1,5]. In the VSIMM methods, the system model sequence, $H(k)$ and the system model set sequence, $S(k)$ are updated at each scan. In this update, only the admissible system models and system model sets are selected. Consider the admissible system model in the variable structure MM. Assume that the system model set sequence, $S(k)$, is an admissible system model set sequence. If $S(k)$ is an admissible system model set sequence, then for every $m_{s(t)}$, we can find at least one $m_{s(t-1)}$ that satisfies $p_{s(t),s(t-1)} \neq 0$ for $t=1,\ldots,k$. Assume the system model sequence, $H(k)$, should also be an admissible system model sequence. If $H(k) = \{m(1), \ldots, m(t-1), m(t), \ldots, m(k)\}$ is an admissible system model sequence, then $p_{s(t),s(t-1)} \neq 0$ for $t=2,\ldots,k$.

3.4.3 The System Model Selection Methods

Three system model selection methods presented in [1,5,14]: Active Digraph (AD), Digraph Switching (DS), and Adaptive Grid (AG).

In the AD and DS methods, we select a set of system models. In the AD methods, the system models are chosen based on their likelihood values. The predefined number of system models are selected and used in the state estimation. In the DS methods, the geometry of the selected system model set is predefined. We compare the overall likelihood values of the predefined system model sets and choose the system model sets for the state estimation.

In contrast to the AD and DS methods, the AG method is built based on the parameter of the system model. The output of the AG methods is to find the key parameter that separating the system models. In the AG methods, if the key parameter is changed, we think the system model is changed. As a result, the system model set is updated when the key parameters of the system models are changed.

3.4.4 Three Parameters for the System Model Estimation in VSIMM

In the equations (3.9), three parameters are required for calculation of the score. The three score is explained based from the practical view point.

The first parameter is the first factor of the right side of the equation (3.9), $p\left(y(k) \mid I_{m_i,m_j}(k), y^{k-1}\right)$, which is the information from the updated measurement. The first parameter is the likelihood of the current measurement, which is calculated based on the current measurement and the prediction of measurement based on the previous system model information. If the measurement and the measurement prediction become increasingly similar, the first parameter value also increases.

The second parameter is the second factor of the equation (3.9), p_{m_i,m_j}, which is defined in the MTPM. This probability value indicates the tendency of the change of the target movement. If a system model moves based on a system model, $m_j(k-1)$, then the system model for target movement at scan k, is predicted to be $m_i(k)$ with probability of the second score. If the second score p_{m_i,m_j} is near 1, the system model of the target should almost change from $m_j(k-1)$ to $m_i(k)$.

The third parameter is the third factor of the equation (3.9), $b_j(k-1)$, which are the past score. In Markov jump process, we should know the most recent information in order to know the current information. The third parameter is the most recent information in Markov jump process. If one of the three parameters is not correct, the state estimation is erroneous.

4. Multiple Model Method with Neural Network}

In this section, a problem in the dynamic MM is explained and analyzed. Based on the problem, a solution is proposed. The solution is built based on the neural network.

4.1 System Model Jump Delay

4.1.1 Background

In the maneuvering target tracking applications, it is difficult to know the exact system model transition information - in this case, the second parameter value is not known. In order to obtain the exact system model transition information, the designer should know the MTPM for the specific maneuvering motion in the application. However, the maneuvering motion only depends on the target. The target decides everything related to maneuvering motion such as when it starts/stops maneuvering, how rapidly it maneuvers, how frequently it maneuvers, and so on. In this case, the designer can not know the probability that the target changes its own motion from one system model to the other system model. As a result, it is difficult to know the MTPM in the real application. However, in the VSIMM, the MTPM is set by the designer's value, which is defined by the designer before target tracking. In this section, the problem in the dynamic MM is explained and analyzed.

4.1.2 Models for System Model Transition in the Dynamic MM

Assume that a system model set M is prepared. All elements of M are connected with system model transition probability. Every system model is connected at least one model except itself with non-zero system model transition probability.

Definition 1: Direct Distance from m_j to m_i, $i \neq j$

The direct distance of two system model from m_j to m_i is defined as $dd(m_i, m_j) = 0$ for $p_{m_i, m_j} = 0$ and $dd(m_i, m_j) = 1$ for $p_{m_i, m_j} \neq 0$.

Definition 2: Shortest Transition Path from m_j to m_i, $i \neq j$

The shortest path form from m_j to m_i, $J(m_i, m_j)$ is defined as the system model jump sequence with the smallest number of jumps from m_j to m_i. A system model m_1 can only jump to a system model m_2 when $p_{m_1, m_2} \neq 0$. The $J(m_i, m_j)$ includes the stating system model m_i and the destination system model m_i.

Definition 3: The Length of Shortest Transition Path from m_j to m_i, $i \neq j$

The length of the shortest path $J(m_i, m_j)$, $L(m_i, m_j)$, defines as *l-1*, where *l* is the number of elements in $J(m_i, m_j)$.

Definition 4: Indirect Distance from m_j to m_i, $i \neq j$

The indirect distance of two system model, $id(m_i, m_j)$, is defined as $id(m_i, m_j) = L(J(m_i, m_j))$.

Example 1

The Figure 2 shows an example of system model set for a dynamic MM. The MTPM for The figure 2 is given by

$$\begin{bmatrix} p_{A,A}=0.5 & p_{B,A}=0.5 & p_{C,A}=0 & p_{D,A}=0 & p_{E,A}=0 \\ p_{A,B}=0.25 & p_{B,B}=0.5 & p_{C,B}=0.25 & p_{D,B}=0 & p_{E,B}=0 \\ p_{A,C}=0 & p_{B,C}=0.25 & p_{C,C}=0.5 & p_{D,C}=0.25 & p_{E,C}=0 \\ p_{A,D}=0 & p_{B,D}=0 & p_{C,D}=0.25 & p_{D,D}=0.5 & p_{E,D}=0.25 \\ p_{A,E}=0 & p_{B,E}=0 & p_{C,E}=0 & p_{D,E}=0.5 & p_{E,E}=0.5 \end{bmatrix} \quad (4.1)$$

First, check the direct distance, shortest path, and indirect distance of 1) A from B 2) from A to D. First, the direct distance is given by $dd(A,B)=1$ and $dd(A,D)=0$. Second, the shortest path is given by $J(A,B)=\{A,B\}$\$ and $J(D,A)=\{A,B,C,D\}$. Finally, the indirect distance is given by $id(A,B)=J(B,A)=1$ and $id(D,A)=J(D,A)=3$.

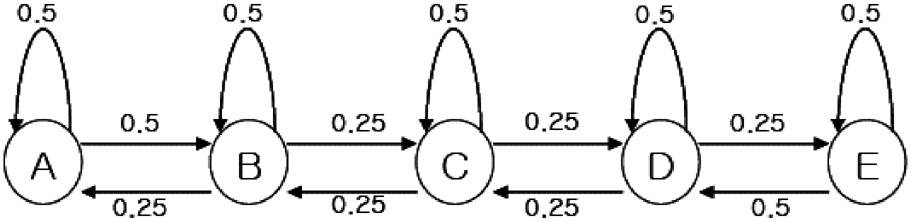


Figure 2 A System Model Set with Transition Probability

4.1.3 A Problem in the Dynamic MM

In the dynamic MM methods, the system model transition is restricted due to the MTPM. In the IMM, $p_{m_i,m_j}=0$ means that the direct system transition from m_j to m_i is impossible. In the VSIMM, $p_{m_i,m_j}=0$ means that m_i is not admissible system model of m_j - m_i is excluded from the candidate system model set for the state estimation.

Theorem: If a target changes its dynamic model and the indirect distance from previous to the current system model of the target is greater than 1, the system model estimation cannot give the true current system model for estimation.

Proof: Assume that the system model jump from $m_j(k-1)$ to $m_i(k)$ when a target maneuvers. The score equations (3.4) and (3.9) have the term p_{m_i,m_j}, the Markov transition probability. If $id(m_i, m_j) > 1$, then the $p_{m_i,m_j} = 0$. As a result, the score of $m_i(k)$ is zero and the current real system model m_i can not be selected.

The indirect distance from a system model to a system model is defined based on the Markov transition probability. If the designer does not know the exact value of the Markov transition probability, then the tracking can not be optimal. However, it is impossible for designer to know the exact value of the Markov transition probability.

Example 2

Consider the system model set in the Figure 2. A target is moving with system model A and suddenly changes its system model to D. The direct system model jump from D to A is impossible due to $p_{A,D} = 0$.

4.1.4 System Model Jump Delay

The unknown MTPM causes the error in the state estimate. With wrong second parameter, the score of the system model is not correct. Now the situation after selecting wrong second parameter is explained. Although the system model estimation is not correct due to the wrong MTPM at the beginning of maneuvering, the system model estimation is being improved more and more as time goes on. Assume that the target starts maneuvering and the system model of the target is changed from $m_j(k-1)$ to $m_i(k)$. In this case, the first parameter value of $m_i(k)$ should be the largest. However, the MPTM is not well-designed for the system model jump from $m_j(k-1)$ to $m_i(k)$ such that $p_{m_i,m_j} = 0$ or $p_{m_i,m_j} \approx 0$. In this case, since the small p_{m_i,m_j} decreases the second parameter, the score is predicted to be small. Next, consider the score of $m_j(k-1)$. The first parameter value is small, since the true system model is changed from m_j to m_i. The small first sub-score makes the score of m_i to be small. As a result, it is predicted that the system model set that includes neither $m_i(k)$ nor $m_j(k)$, are selected for the state estimation. Assume that the system model set that includes $m_l(k)$ is selected for the state estimation. In this case, the system model estimation is not correct at the beginning of maneuvering, since $m_l(k)$ is selected instead of the true system model $m_i(k)$. However, if $p_{m_l,m_j} >> p_{m_i,m_j}$ and the first parameter is kept largest, then the score of $m_i(k+1)$ increases. As time goes on, the true system model, m_i is selected for the state estimation. In this case, the system model estimator needs additional scans to find the true system model. In this dissertation, it is call the system model jump delay.

Example 3

Consider the system model set in the Figure 2. When a target changes its system model from A to *D*, it is impossible to select *D* in the dynamic MM. However, it is possible that the system model changes through the shortest path, $J(D,A)=\{A,B,C,D\}$. At the next scan after the maneuvering, it is possible for the system model B to be selected. In this case, $id(D,A)=3$ and more scans are needed. Next scan, the system model *B* can be selected. After the system model *B* is selected, the next model to jump is the system model *C*. At the next scan, the system model D is predicted to be selected, which is the true system model for state estimation. As a result, at least, three scans are required for searching the true target. In other words, more than three scans are required.

4.1.5 A Solution: Neural Networks

In the dynamic MM, the MPTM is selected by designer. Although the MPTM is not generally known, the designer determined its value based on the designer's experience or trial and error. As a result, the tracking error increases due to the inexact MTPM. Therefore, it is required to eliminate Markov jump process in the dynamic MM. In order to reduce the system model jump delay, the Markov jump process is eliminated.

Instead of Markov jump process, neural network gives the system model transition information. An alternative method is needed for the MTPM. The MTPM has the system model transition information. As a result, the alternative method should have a function that extracts the system model transition information. The neural network is a method for searching recognition and classification. The suitable system model for the current situation is found through the searching function of neural network.

In this project, a new system model estimation method using neural network is proposed instead of (3.9) and (3.12).

4.2 Neural Network in the Dynamic MM

In this subsection, the process of adopting neural network into the dynamic MM is proposed. First, the role of the neural network is explained. Second, the neural network is designed for the dynamic MM. Third, the neural network training is explained. Lastly, the advantages in adopting neural network are presented by comparing the VSIMM.

4.2.1 Assumptions- Prior Knowledge on the Target Movements

Before designing the system model estimation method, we make two assumptions on target movements.

The first assumption is that we know the limit of target movements. The target decides everything on its own movement but cannot make a movement that is over its movement limit. For an extreme example, The airplane cannot suddenly stop and keep its position.

The second assumption is that we know the representative moving patterns of target. For example, a car on the cross has the four possible moving patterns- 1) turn left, 2) turn right, 3) go straight, and 4) stay.

The first assumption is reasonable. If we do not know the limit of the target movements, we can not design the system models for the unlimited movements. The system models are defined based on the unknown target movement parameters such as the maximum acceleration and the maximum turn rate.

In order for the second assumption, the representative trajectories of the target are required. A trajectory of the target implies the moving patterns. From the past movement, we can know the movement patterns of the target.

4.2.1 Role of Neural Network

Consider the backpropagation (BP) neural network [15, 16]. The backpropagation algorithm is for a multiple layer neural network, where a complex task can be learned [15, 16]. Since the information of the system model transition is hidden, implicit, and complex, the backpropagation network is chosen.

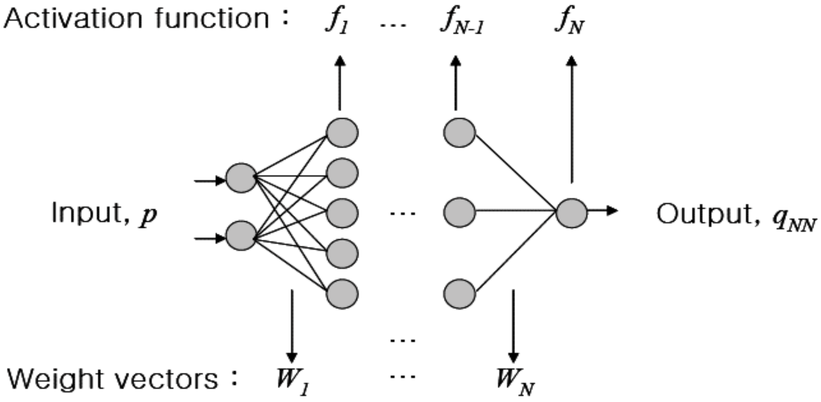


Figure 3 Backpropagation Neural Network

Consider a BP neural network with N layers. The BP network has N weight matrices, $\{W_i\}$ $i=1,\ldots,N$. Each layer has a activation function, f_i for $i=1,\ldots,N$. Let p denote the input of the BP network. The output of the BP network, q_{NN} is given by

$$q_{NN} = f_N\left(W_N f_{N-1}\left(\mathrm{K}\, W_2 f_1\left(W_1 p\right)\right)\right) \tag{4.2}$$

Figure 3 shows the BP network.

The objective of neural network is to find a desired output q_D for an input p. The pair $\{p, q_D\}$ is called training pair. Suppose that we have L training pairs, $\{p_1, q_{D,1}\},\mathrm{K}\ ,\{p_L, q_{D,L}\}$. The cost function of the BP network, *J* is given by

$$J = \sum_{i=1}^{L}\left(q_{D,i} - q_{NN,i}\right)^T\left(q_{D,i} - q_{NN,i}\right), \tag{4.3}$$

where $q_{NN,i}$ is the output of the BP neural network for input, p_i. We determine the weight matrices, $W_1,\mathrm{K}\ ,W_N$ that minimize the cost function *J*, which is given by

$$J \xrightarrow[W_1,\mathrm{K}\,,W_N]{} \min \tag{4.4}$$

This is a kind of least square estimation [17] in deterministic sense. The process of determining the weight matrices is called neural network training. The neural network is trained in iterative way. The weight matrices are updated and we calculate the error between the desired output and the output using the updated weight matrices, $e_T = q_D - q_{NN}$. This process is repeated until the stop condition of training is satisfied. In general, there are two stop conditions. One is that $|e_T|$ is within the predefined tolerance e , that is, $|e_T| < e$, and the other is that the number of iterations are over the predefined maximum number of iterations.

4.2.2 Neural Network Based Dynamic MM

To reduce system model jump delay, the neural network is used instead of Markov jump process. The equations (3.1) is rewritten as

$$\hat{x}(k) = \sum_{i=1}^{n} f_{NN,i}(k)\hat{x}_i(k) \tag{4.5}$$

where $f_{NN,i}$ is the score of m_i that is implemented based on the neural network.

The $f_{NN,i}$ is implemented by using the neural network. In the equation (4.5), the neural network gives the score of the system model. In the previous section, the scores of the system models are calculated in the recursive form based on the Markov jump process.

The inputs of the $f_{NN,i}$ are the first and the third parameters described in the subsection 3.4.4. The first parameter is the updated information from the new measurement, $I_{updated}$, which is not changed due to the design of the system models. The third parameter is the information of the selected system models at the previous scan denoted by $I_{previous}$. The score of a system model $m_i(k)$ at scan k is given as

$$b_i = f_{NN,i}(I_{updated}, I_{previous}). \tag{4.6}$$

The cost function of the neural network for the dynamic MM, J_{DynMM}, is deigned as

$$J_{DynMM} = \left(x(k) - \sum_{i=1}^{n} f_{NN,i}(k)\hat{x}_i(k) \right)^2. \tag{4.7}$$

The objective is to minimize the J_{DynMM} with respect to the weight matrices of neural network, $W_1, \mathrm{K}\ , W_N$, which is given as

$$J_{DynMM} \xrightarrow[W_1,\mathrm{K}\ ,W_N]{} \min\ . \tag{4.8}$$

4.2.3 Neural Network Design

The neural net design is the process of determining the inputs, outputs of the neural network and the structure of neural network.

In the neural network design, the system model set is the key element. In the VSIMM, three methods for system model set grouping was proposed [1, 5, 14] and briefly described in the subsection 3.4.3. Three different methods generates the different system model set deign.

In AD and DS methods, the system models are designed in discrete manner. In these cases, the system models are defined in the discrete set. For the AD methods, the system models with highest scores are selected. For the DS methods, the system model group with the highest score is selected. In the DS methods, the geometry of the selected system model group is fixed and the scores of groups are compared but in the AD methods, only the system models with highest scores are selected without considering the geometry of the selected system model group.

In the AG methods, the system models are defined in the continuous space. The distance between the neighbor system models are variable. The spacing process is adaptively carried out The AG methods seem more flexible than the AD and the DS methods.

The information type of the selected system model depends on the system model design. Different types of the system models cause the different types of the neural network inputs and outputs.

The system model set selection method described in the subsection 3.4.3 is taken into account in designing inputs and outputs of neural network. If the AG method is used in the system model selection, we treat the system model in the continuous space. If the AD or DS method is used in the system model selection, we treat the model in the discrete space.

The design methods of the inputs and the outputs of $f_{NN,i}$ are divided into two methods: model based design and parameter based design.

Model Based Design

The model based design is focused on not the key parameter but the system model, which is similar to the AD and the DS methods in VSIMM. The system models are designed in discrete manner and the number of system models is finite. In this case, the neural network selects the system model for the state estimation. The input to $f_{NN,i}$ is defined by $p_{\mathrm{mod}\,el} = \{I_{updated}, m(k-1)\}$. The output of $f_{NN,i}$ is given by $q_{\mathrm{mod}\,el} = \{b_{m_i(k)}\}$.

Parameter Based Design

In the parameter based design, the output of neural network is the key parameter that separating the system models. In this case, the input to $f_{NN,i}$ is given by $p_{para} = \{I_{updated}, q(k-1)\}$, where $q(k-1)$ is the key parameter of the selected system model at scan $k-1$. The output of $f_{NN,i}$ is given by $q_{paral} = \{b_{q(k)}\}$.

4.2.4 Neural Network Training

The neural network training is to find the weight matrices of the neural network, $W_1, \ldots, W_N$, that minimize the equation (4.8).

Prepare the Training Trajectories

The first step of neural network training is to prepare the training trajectories. In the equation (4.7), the squared error, $e_{NN}^2 = \left(x(k) - \sum_{i=1}^{n} f_{NN,i}(k)\hat{x}_i(k)\right)^2$, is to be minimized. In the error e_{NN}, the true but unknown value, $x(k)$, should be known. The value $x(k)$ is obtained from the training trajectories.

For generating the training pairs, training trajectories that contains the representative moving patterns of the target is required. If the information in the training scenarios is not enough to cover the various maneuvering movements, then the state estimation becomes erroneous.

The number of training pair should exceed the number of the total weights in the BP neural network [18].

Simulation of Each System Model by Using the Training Scenarios

In the second step, the error e_{NN} for training pairs are calculated. Prepare the filter bank: one filter is one system model. After the simulations of all filters are carried out and $\hat{x}_i(k)$ for *i=1,…,n* are obtained.

Training Pair Generation

After the simulation, the training pair is extracted from simulation data. For all system models, the e_{NN} can be calculated. The training pair is selected based on e_{NN}. This is a searching process. The system models with small e_{NN} are searched and selected.

The training pair depends also on the system model design. If the AG method is used, the training pair is described in the continuous space. If the AD (or DS) method is used, the training pair is described in the discrete space.

Neural Network Training

In the last step, neural network is trained based on the training pair. In general, there are two termination conditions: e_T and the number of maximum training-iterations. The term e_T is related to the error performance. The limit of iteration is related to the training time. If the weight change is small and the error e_T converges, two cases are predicted. The first case is that the weight is fixed with global minimum of the cost function. The other case is that the weight is in the local minimum. In other words, although the error of the neural network train converges, the neural network is not optimal for the cost function [18]. Therefore, in the neural network training, both e_T and the number of maximum training iterations should be checked. The neural network training process is shown in Figure 4.

Consider the meaning of neural network training. In the dynamic MM with Markov jump process, system model estimation is performed based on the Bayesian approach. The system model probability is recursively updated. All information except for the MTPM is easily obtained. Although the MTPM is not available in system model estimation, the MTPM is also predefined or estimated. However, in the system model estimation using neural network, Markov jump process is eliminated. This means that the additional information corresponding to the MTPM is obtained through the neural network training. The training data reflect priori information of the target movement to the system model estimation, instead of the MPTM.

In the neural network training process, it is the most important to design the training trajectories. The training trajectory depends on the application. For example, the moving patterns of an airplane and a submarine are different. For each case, we find or design the trajectory that includes the representative movements of the target.

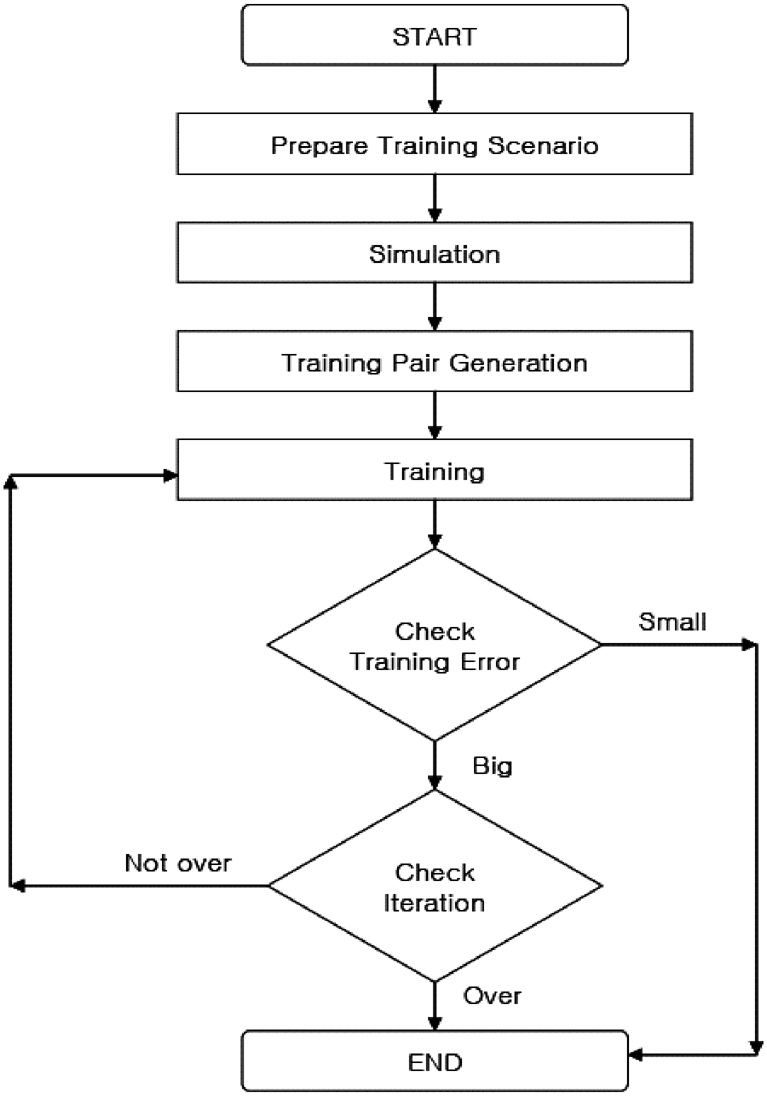


Figure 4 The Neural Network Training Process

4.2.4 Neural Network Structure

The multiple layer BP network can approximate almost any function if the enough hidden layer is prepared [18]. Generally, however, it is difficult to give the number of hidden layers and the number of the neurons in the hidden layer [18]. Other neural network such as Hopfield network and bidirectional associative memory is known approximately [19].

A guide line for the BP neural network structure is that the number of parameters that is changed during the neural network training, such as weight, should fewer than the number of training pairs [18]. The other guide line is that that too small number of neurons in hidden layer cannot approximate the complex function [18]. As a result, the number of hidden layers and the number of neurons are selected by checking the e_{NN}.

4.2.5 Neural Network and VSIMM}

The advantage of the VSIMM methods is that the selected system models are used in the state estimation. The proposed method is applicable to variable structure MM methods by adding the system model selection process. If the system models are selected based only on the scores, it is similar to the AD method. If the system models are selected based on the system model group whose geometry is predefined and fixed, it is similar to the DS method. If the system model is updated with adaptive manner in the continuous space, it is similar to the AG method.

4.3 Discussions

Consider the situation when a target is moving with a untrained moving pattern. An attractive property of neural network is that, for the untrained input, the neural network gives the output of the input that is the most similar to the untrained input. Due to this property, the neural network is adaptive for missing and noisy data [15, 16, 19]. If the untrained moving pattern happens, the neural net gives the system model that is the most similar in the trained patterns.

5. Simulations and Results

In this section, two simulations are shown to verify the proposed method.

5.1 Simulation Model

In both simulations, an object moving in 2-dimensional space is tracked.

5.1.1 Plant and Measurement Equations

The piecewise constant white acceleration model [4] is used for the Kalman filter. The state vector is defined as $x = [x_1, \dot{x}_1, x_2, \dot{x}_2]^T$. The measurement vector is given by $y = [y_1, y_2]^T$. The system noise vector is $w = [w_1, w_2]^T$ and the measurement noise vector is $v = [v_1, v_2]^T$. The system and the measurement models are given by

$$x(k+1) = \begin{bmatrix} F & O_2 \\ O_2 & F \end{bmatrix} x(k) + \begin{bmatrix} \Gamma w_1(k) \\ \Gamma w_2(k) \end{bmatrix} \tag{5.1}$$

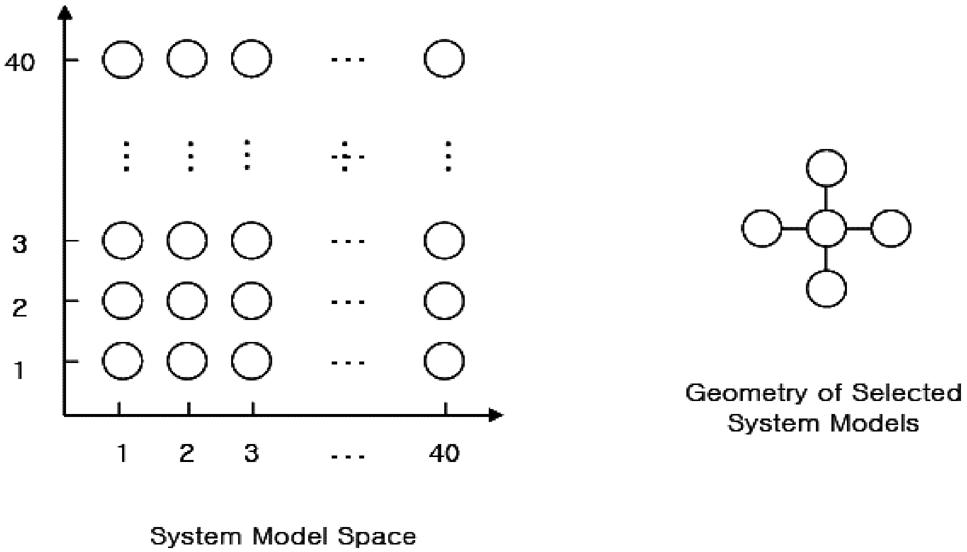
$$y(k) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} x(k) + v(k) \tag{5.2}$$

where $F = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}, O_2 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, \ \Gamma = \begin{bmatrix} \frac{1}{2}\Delta t^2 \\ \Delta t \end{bmatrix}$ and Δt is the constant scan interval. The measurement noise covariance matrix is given by $R_v = I_2$, $I_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$. The system noise block covariance is given by $Q_w = \begin{bmatrix} w_1 I_2 & O_2 \\ O_2 & w_e I_2 \end{bmatrix}$.

5.1.2 System Model Design

The system model is defined based on the noise variances of $w_1(k)$ and $w_2(k)$, $s_{w_1}^2$ and $s_{w_2}^2$. The ranges of $s_{w_1}^2$ and $s_{w_2}^2$ are given by $1 \le s_{w_1}^2, s_{w_2}^2 \le 196$. In both simulations, the system model $m_{i,j}(k)$ is defined by $m_{i,j}(k) \equiv \{s_{w_1}^2 = i, s_{w_2}^2 = j\}$ at scan k.

On DS VSIMM



System Model Space

Geometry of Selected System Models

Figure 5 System Model Space and Geometry of the Selected System Models in the DS VSIMM Simulations

In the DS VSIMM, the total system model set is defined by $M_{VSMM} = \{m_{i,j} \mid i = 5a - 4, j = 5b - 4\}$ for $a,b = 1,K ,40$. Let $m_{i,j}^N$ denote the neighbor models of $m_{i,j}$. In general, the $m_{i,j}^N$ are composed of 4 system models - $m_{i-1,j}$, $m_{i+1,j}$, $m_{i,j-1}$, and $m_{i,j+1}$. For some specific models such as $m_{1,j}$, $m_{40,j}$, $m_{j,1}$, and $m_{j,40}$, *j=2,…,39*, there are three neighbor models and for $m_{1,1}$, $m_{1,40}$, $m_{40,1}$, and $m_{40,40}$, there are only two neighbor models. The system models for state estimation are determined by choosing one system model named the center model. After the selection of the center model, neighbors are determined based on the location of center model. Figure 5 shows the system model space and geometry of the selected system models of the in this simulation DS VSIMM simulations. The MTPM is given by $p_{m_{i,j},m_{i,j}} = 0.6$ and $p_{m^N_{i,j},m_{i,j}} = (1-p_{m_{i,j},m_{i,j}})/n_{i,j}$, where $n_{i,j}$ is the number of neighbors of $m_{i,j}$. The system models whose center model has the maximum score are selected.

On Neural Network

For the proposed method, only on system model is used in the state estimation. In other words, only one system model is selected in the system model estimation. The value of the noise variances of w_1 and w_2 is not the index but the real value given by $1 \le s^2_{w_1}, s^2_{w_2} \le 196$. The inputs of the BP network at time k are the previously selected system model and the likelihood of the current measurement based on the previously selected system model. The currently selected model is the last parameter and the likelihood value is the first parameter at the next scan. The neural networks used in both simulations are the BP network with one hidden layer with 16 neurons.

5.2 Simulation I

5.2.1 Introduction

The Objectives of the simulation I are given as follows.

- Show how to build the proposed method.
- Show the system model jump delay in DS VSIMM.
- Verify the proposed method.

In the simulation, the system model jump delay in DS VSIMM is shown by checking the transition history of the center system model. In order to verify the proposed method, the MSE of the proposed method, DS VSIMM [1,5], and the $a - b$ filter [4] are compared.

5.2.2 Neural Network Training

The training scenario is illustrated in Figure 6. The training scenario include the circle movements and line movement. The training information is as follows.

- \item Number of training pairs : 60 pairs
- \item Number of the maximum iteration : 1000 times
- \item Check Point : number of weights 16 < Number of training pairs 60

As discuss in the previous chapter, the number of the parameters in the BP neural network should be fewer than that of training pairs.

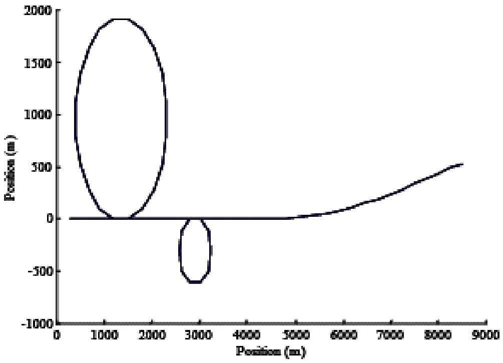


Figure 6 The Training Trajectory in Simulation I

5.2.3 Test Scenario

The test trajectory is shown in Figure 7. In the test scenario, when the target turns, it is predicted that maneuver happens. There are four maneuvering periods: 1) scan 11~13, 2) scan 21~23, 3) scan 31~33 and 4) scan 41~44.

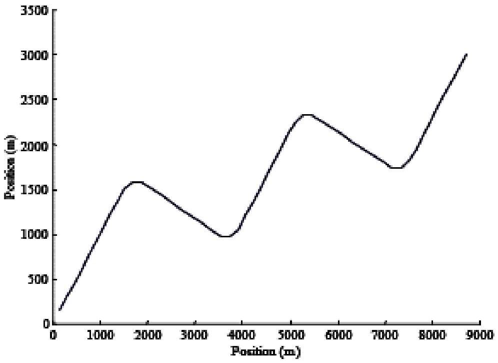


Figure 7 The Test Trajectory in Simulation I

5.2.4 Results

The results are obtained after 50 Monte-Carlo runs.

System Model Jump Delay

Figure 8 shows the history of the selected system model of the proposed and the center system model of the DS VSIMM methods. The system model jumps of NN-VSMM in maneuvering preriods are faster than those of the DS VSIMM. This is due to the system model jump delay.

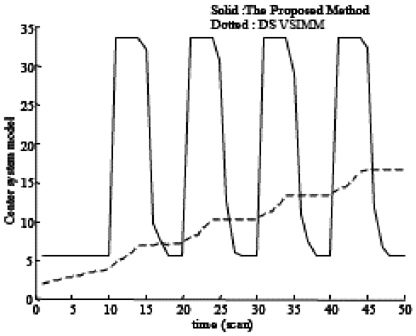


Figure 8 The System Model Jump

Performance Comparison

In Figure 9, the mean square errors (MSE) of the proposed and the DS VSIMM methods are compared. In a non-maneuvering period, from scan 1 to scan 10, the MSE of both methods are similar, but in the maneuvering periods, the proposed method is less erroneous than the DS VSIMM. The MSE in maneuvering periods is given in the table \ref{Tableresut1MSE}. As a result, the system model jump delay by using the proposed method is reduced.

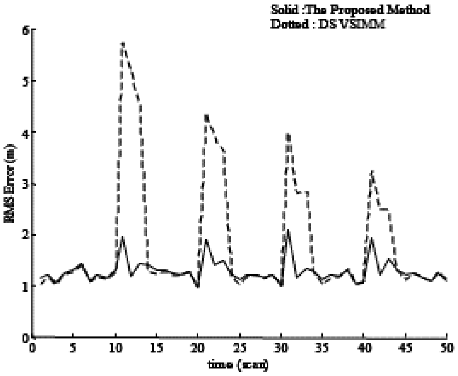


Figure 9 The MSE of the Proposed Method and DS VSIMM

Scan(Maneuvering periods)	The Proposed	DS-VSMM	α-β filter
11-13	1.5791	4.6258	31.0747
21-23	1.6138	3.7418	31.1780
31-33	1.5083	2.9099	30.9601
41-43	1.5944	2.5968	31.1208

Table 1 The MSE in Simulation I

5.3 Simulation II

5.3.1 Introduction

The Objectives of the simulation II are given as follows.

- Show the effect of the training pair.
- Compare the DS VSIMM method and the proposed methods with different training scenarios.

The simulation focuses on the effect from the training scenario. As explained in the previous chapter, the representative training scenario is required. In the simulation, three training scenarios are prepared. First scenario is not sufficient to cover complex movement. Second scenario can covers complex movements. The last scenario is mix of the fist and the second. The performance of the training scenarios and DS VSIMM are compared.

5.3.2 Neural Network Training

Three training scenarios are prepared- 1)Line movements, 2)Circle movements, and 3)Line + Circle movements. The training scenarios are given in Table 2 and Figures 10, 11, and 12.

Line Scenario

The line scenario is the simplest. There is no change of the acceleration. The simulation model in the equations (5.1) and (5.2) is predicted to be covered the movement in line scenario with the small system model jump.

Circle Scenario

The circle scenario is more complex than the line scenario. Moreover, more system jump is predicted than the line scenario.

Line+Circle Scenario

This scenario is the mixture of the line and the circle scenarios. In this simulation, this scenario is mainly compared with the circle scenario.

Training Scenario	Descriptions
Line Scenario	Initial position: (0,0) Velocities : (200, 100), (-200, 100), (200, -100), (-200, -100)
Circle Scenario	Angular velocity: π/10 Radius: 50, 100, 150, 200
Line+Circle Scenario	Line - Initial position: (0,0) Velocities : (200, 100), (-200, -100), Circle - Angular velocity: 10/π , Radius: 50, 100

Table 2 Training Scenarios

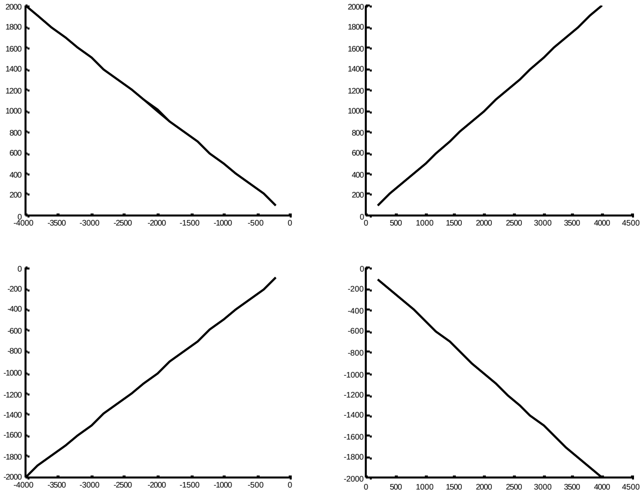


Figure 10 Training Trajectories : Line Scenario

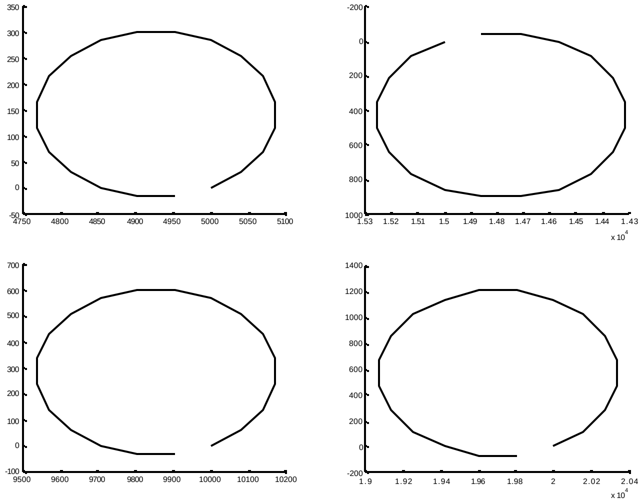


Figure 11 Training Trajectories : Circle Scenario

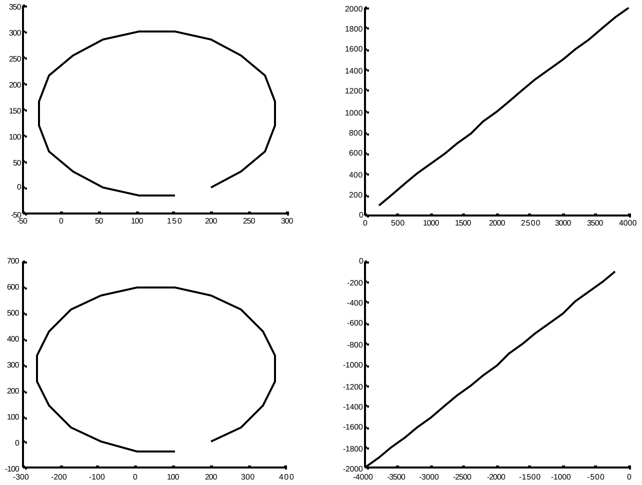


Figure 12 Training Trajectories : Circle+Line Scenario

5.2.3 Test Scenario

The three proposed methods with different training scenarios with the trajectories illustrated in Figures 13, 14, 15, and 16 are used in this simulation. Three scenarios are similar to the training scenarios and one scenario is different.

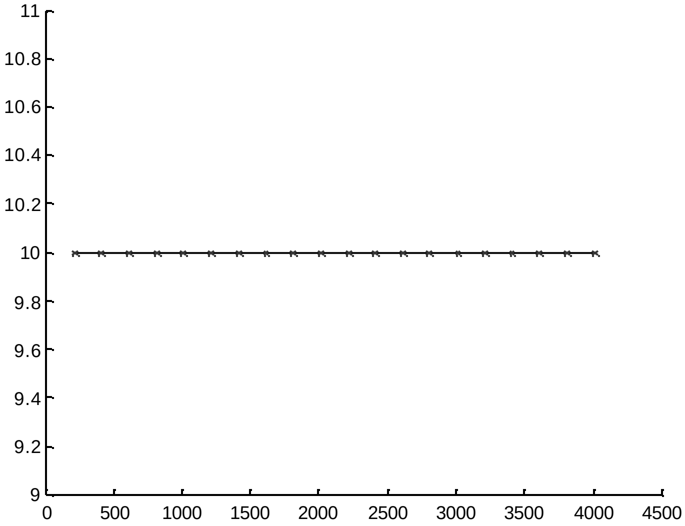


Figure 13 Test Scenario Trajectory: Line

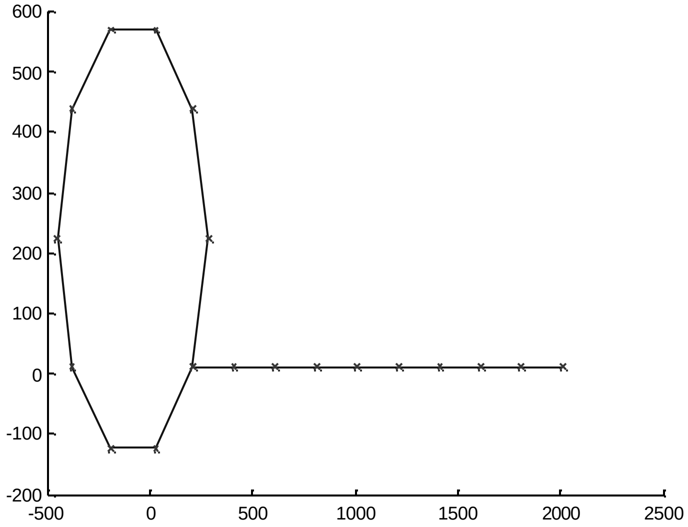


Figure 14 Test Scenario Trajectory: Circle+Line

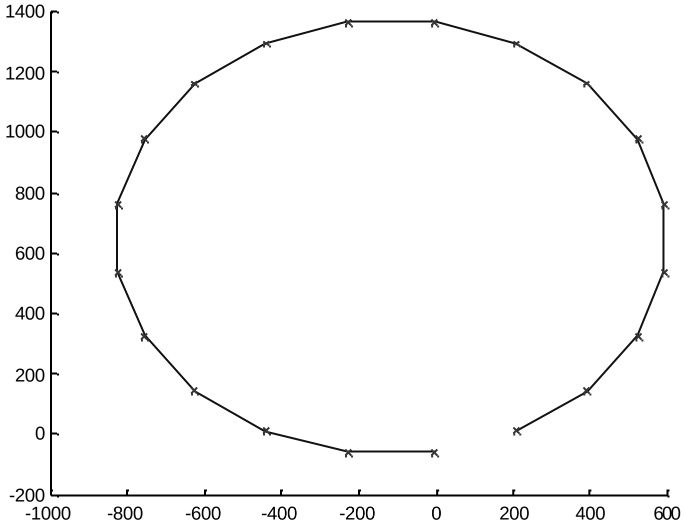


Figure 15 Training Scenario Trajectory: Circle

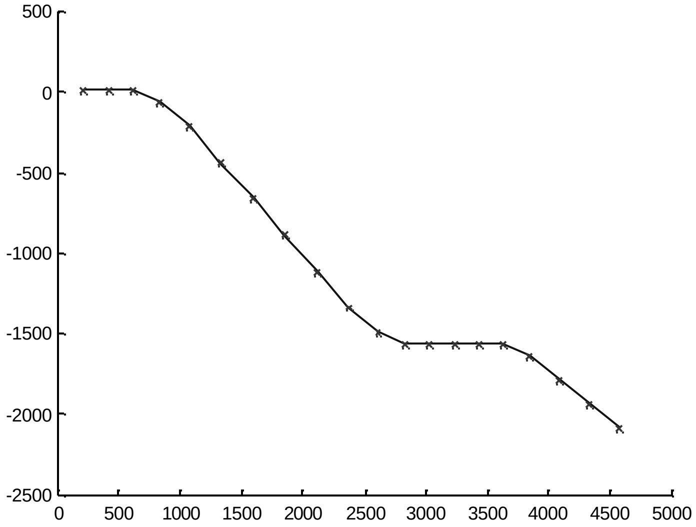


Figure 16 Training Scenario Trajectory: Untrained

5.3.4 Results

The comparison of the MSE of four methods is shown in Table 3.

Test Scenario	Train Scenario Fiters	Line	Circular	Line + Circle
Linear	NNVSMM	1.0301	1.1948	1.1154
	DSVSMM	1.1573	1.1469	1.1472
Circ le	NNVSMM	33.8047	3.1184	2.0502
	DSVSMM	5.9898	5.9647	5.9898
Line+Circle	NNVSMM	23.2437	6.0346	2.8477
	DSVSMM	5.2321	5.2337	5.2366
Untrained	NNVSMM	16.8466	1.5515	1.6559
	DSVSMM	2.5445	2.5178	2.5463

Table 3 The MSE of the test trajectories

Line Test Scenario

The MSE of VSIMM and all proposed methods shows similar performance. Especially, the proposed method with only circle train scenario does not give the erroneous estimation.

Circle Test Scenario

The proposed method with the line train scenario is significantly erroneous. Other proposed methods shows better performance than DS VSIMM. Moreover, the proposed method with the line+circle scenario shows the best performance. The detail comparison is in Figures 17 and 18.

Line+Cicle Test Scenario

The DS VSIMM shows better performance than the proposed method with the circle train scenario. As a result, the selection of training scenario is important.

Unknown Test Scenario

The proposed methods with the circle scenario and the line+circle scenario show better performance than DS VSIMM.

From the result, it is predicted that

- the training scenario should be cover various target movements, and
- for untrained trajectory, the proposed algorithm shows better performance than DS VSIMM.

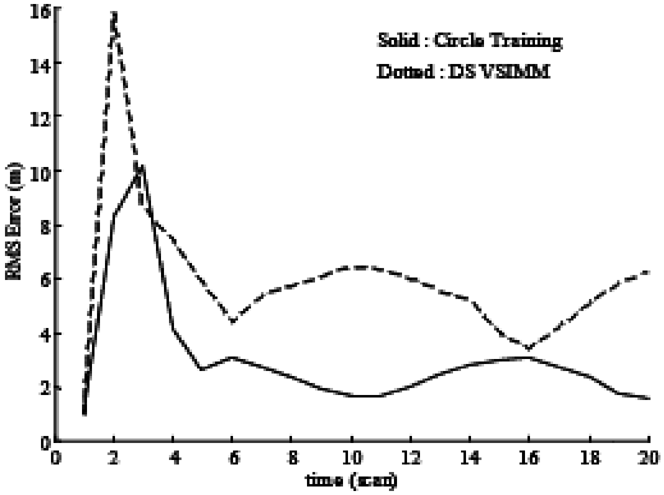


Figure 17 The Result of Circle only Traing

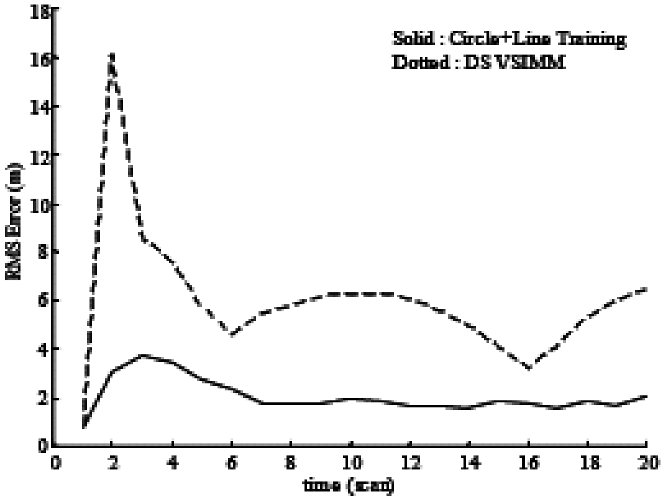


Figure 18 The Result of Circle + Line Train

5.4 Discussions

In Simulation I, the proposed method has less system model jump delay than the DS VSIMM. This is directly connected to the estimation error reduction. In Simulation II, the effect of training scenario is analyzed. The training scenario with only one moving pattern is not suitable for covering the various moving patterns. The Circle + Line training scenario is better than the Circle scenario. Moreover, it is shown that the untrained moving patterns can be dealt with neural network- the MSE of DS VSIMM is the second largest.

6. Conclusion

In this project, a new system model estimation method using neural network has been proposed. Firstly it has been shown that the Markov jump process causes system model jump delay. Instead of Markov jump process, a neural network in system model estimation is employed for reducing the system model jump delay. In representative simulations, it is shown that the reduction of system model delay in the proposed method is achieved. Moreover, the effect of the neural network training scenarios for generality is analyzed. It also is shown that the untrained moving patterns can be dealt with neural network.

References

[1] X. R. Li, “Hybrid Estimation Techniques,” *Control and Dynamic Systems*, **76**, pp.213-287, 1996.

[2] M. S. Grewal and A. P. Andrews, *Kalman Filtering Theory and Practice*, Prentice Hall, New Jersy, 1993.

[3] S. Blackman and R. Popoli, *Design and Analysis of Modern Tracking Systems*, Artech House, Boston, 1999.

[4] Y. Bar-Shalom and X. R. Li, *Estimation and Tracking Principles, Techniques, qnd Software*, Artech House, Boston, 1993.

[5] X. R. Li and Y. Bar-Shalom, “Multiple model estimation with variable structure”, *IEEE Transactions on Automatic Control*, **41**, No. 4, pp.478-493, Apr. 1996.

[6] A. Papoulis, *Probability, Random Variables, and Stochastic Processes, 3rd Ed.*, McGraw-Hill, New York, 1991.

[7] S. Shams, “Neural Network Optimization for Multi-Target Multi-Sensor Passive Tracking”, *Proceeding of The IEEE*, **84**, No.10, pp.1442-1457, October 1996.

[8] L. Chin, “Application of Neural Networks in Target Tracking Data Fusion”, *IEEE Transactions on Aerospace and Electronic Systems*, **30**, No.1, pp.281-287, Jan. 1994.

[9] S. Silven, “A Neural Approach to the Assignment Algorithm for Multi-Target Tracking”, *IEEE Transactions on Oceanic Engineering*, **17**, No.4, pp.326-332, October 1992.

[10] B. Zhou, “A Comprehensive Analysis of Neural Solution to the Multitarget Tracking Data Association Problem”, *IEEE Transactions on Aerospace and Electronic Systems*, **29**, No.1, pp.260-263, Jan. 1993.

[11] P. J. Pacini and B. Kosko, “Adaptive Fuzzy Systems for Target Tracking”, *Intelliget System Engineering*, pp.3-21, Autumn 1992.

[12] Kyung-Ho Cho, Han-Seok Ko, Byung-Ha Ahn, “Intelligent Adaptive Gain Adjustment and Error Compensation for Improved Tracking Performance”, *IEICE Trans. Fundamentals*, **E83-D**, No.11, pp.1952-1959, November 2000.

[13] A. Doucet, and R. Ristic, “Recursive state estimation for multiple switching models with unknown transition pprobabilty”, *IEEE Transactions on Aerospace and Electronic Systems*, **38**, No.3, pp.1098-1104, July 2002.

[14] V. P. Jilkov, D. S. Angelova, TZ. A. Semirdjief, “Design and Comparison of Mode-Set Adaptive IMM Algorithms for Maneuvering Target Tracking”', *IEEE Transactions on Aerospace and Electronic Systems*, **35**, No. 1, pp.343-350, Jan. 1999.

[15] P. D. Wasserman, *Neural Computing: Theory and practice*, Van Nostrand Reinhold, New York, 1989.

[16] S. Haykin, *Neural Networks: A Comprehensive Foundation*, Macmillan College Publishing Company, New York, 1994.

[17] S. M. Kay, *Fundamentals of Statistical Signal Processing, vol. 1 Estimation Theory*, Prentice Hall, New Jersy, 1993.

[18] M. T. Hagan, H. B. Demuth, M. Beale, *Neural Network Design*, PWS Publishing Company, Boston, 1996.

[19] L.V. Fausett, *Fundamentals of Neural Networks: Architectures, Algorithms and Applications*, Prentice Hall, 1994.